CLAIMS

I claim:

1. A computer readable medium programmed to operate a method of removing an empty string term from an automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the automaton A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition,

wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the automaton A.

2. The computer readable medium of claim 1, the method on the computer readable medium further comprising:

removing inaccessible states using a depth-first search of the automaton A.

3. The computer readable medium of claim 1, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

4. The computer readable medium of claim 1, wherein the step of computing ε-closure for each input state of an input automaton "A" further comprises:

removing all transitions not labeled with an empty string from automaton A to produce an automaton A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

10. A circuit programmed to operate a method of removing an empty string term from an automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the automaton A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition,

wherein each state "q" is left with its weights pre-multiplied by an ε-distance from state "p" to a state "q" in the automaton A.

11. The circuit of claim 10, the method programmed into the circuit further comprising:

removing inaccessible states using a depth-first search of the automaton A.

12. The circuit of claim 10, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights $(d[p,q] \otimes w[e])$ to the transitions leaving p.

13. The circuit of claim 10, wherein the step of computing ε-closure for each input state of an input automaton "A" further comprises:

removing all transitions not labeled with an empty string from automaton A to produce an automaton A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

14. A computer readable medium programmed to operate a method of removing an empty string term from a transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the transducer A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition,

wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the transducer A.

15. The computer readable medium of claim 14, the method on the computer readable medium further comprising:

removing inaccessible states using a depth-first search of the transducer A.

16. The computer readable medium of claim 14, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

17. The computer readable medium of claim 14, wherein the step of computing ε-closure for each input state of an input transducer "A" further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

18. A circuit programmed to operate a method of removing an empty string term from a transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the transducer A;

modifying outgoing transitions of each state "p" by:

removing each transition labeled with an empty string; and

adding to each transition leaving "p" a non-empty-string transition,

wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the transducer A.

19. The circuit of claim 18, the method programmed into the circuit further comprising:

removing inaccessible states using a depth-first search of the transducer A.

20. The circuit of claim 18, wherein the step of adding to E[p] non-empty-string transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

21. The circuit of claim 18, wherein the step of computing ε-closure for each input state of an input transducer "A" further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

22. An automaton B having no ε-transitions, the automaton B produced according to a method of removing the ε-transitions from an input automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input automaton;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the input automaton A.

22. The automaton of claim 21, the method of creating the automaton B further comprising:

removing inaccessible states using a depth-first search of the input automaton.

23. The automaton of claim 21, wherein the step of adding to E[p] non-ε-transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

24. A automaton of claim 21, wherein the step of computing an ε-closure for each input state of an input automaton A further comprises:

removing all transitions not labeled with an empty string from automaton A to produce an automaton A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

25. A transducer B having no ε-transitions, the transducer B produced according to a method of removing the ε-transitions from an input transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input transducer;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-⊗-multiplied by an ε-distance from state "p" to a state "q" in the input transducer A.

26. The automaton of claim 25, the method of creating the transducer B further comprising:

removing inaccessible states using a depth-first search of the input transducer.

27. The automaton of claim 25, wherein the step of adding to E[p] non-ε-transitions further comprises leaving q with weights (d[p,q] ⊗ w[e]) to the transitions leaving p.

28. A automaton of claim 25, wherein the step of computing an ε-closure for each input state of an input transducer A further comprises:

removing all transitions not labeled with an empty string from transducer A to produce a transducer A_ε;

decomposing A_ε into its strongly connected components; and

computing all-pairs shortest distances in each component visited in reverse topological order.

29. A computer readable medium storing an executable automaton B having no ε-transitions, the automaton B produced according to a method of removing ε-transitions from an input automaton A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input automaton;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-multiplied by an ε-distance from state "p" to a state "q" in the input automaton.

30. A computer readable medium storing an executable transducer B having no ε-transitions, the transducer B produced according to a method of removing ε-transitions from an input transducer A having a set of states "p" and a set of states "q", the method comprising:

computing an ε-closure for each state "p" of the input automaton;

modifying outgoing transitions of each state "p" by:

removing each ε-transitions; and

adding to each transition leaving "p" a non-ε-transitions, wherein each state "q" is left with its weights pre-multiplied by an ε-distance from state "p" to a state "q" in the input transducer.